Project: Wrangling and Analyze Data

Data Gathering

In the cell below, gather **all** three pieces of data for this project and load them in the notebook. **Note:** the methods required to gather each data are different.

1. Directly download the WeRateDogs Twitter archive data (twitter_archive_enhanced.csv)

In [2]:
```
#import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import requests
import json
import tweepy
```

In [3]:
```
t_archive=pd.read_csv(r"C:\Users\HP\.jupyter\dogs\twitter-archive-enhanced.csv")
t_archive.head()
```

Out[3]:

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	sour
0	892420643555336193	NaN	NaN	2017-08-01 16:23:56 +0000	href="http://twitter.com/download/iphon
1	892177421306343426	NaN	NaN	2017-08-01 00:17:27 +0000	href="http://twitter.com/download/iphon
2	891815181378084864	NaN	NaN	2017-07-31 00:18:03 +0000	href="http://twitter.com/download/iphon
3	891689557279858688	NaN	NaN	2017-07-30 15:58:51 +0000	href="http://twitter.com/download/iphon
4	891327558926688256	NaN	NaN	2017-07-29 16:00:24 +0000	href="http://twitter.com/download/iphon

1. Use the Requests library to download the tweet image prediction (image_predictions.tsv)

In [4]:
```
url='https://d17h27t6h515a5.cloudfront.net/topher/2017/August/599fd2ad_image-predictions
```

```
response=requests.get(url)

with open('image_predictions.tsv',mode='wb') as file:
    file.write(response.content)
```

In [5]:
```
image_predictions=pd.read_csv('image_predictions.tsv',sep='\t')
image_predictions.head()
```

Out[5]:

	tweet_id	jpg_url	img_num	p1	p1_co
0	666020888022790149	https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg	1	Welsh_springer_spaniel	0.4650
1	666029285002620928	https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg	1	redbone	0.5068
2	666033412701032449	https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg	1	German_shepherd	0.5964
3	666044226329800704	https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg	1	Rhodesian_ridgeback	0.4081
4	666049248165822465	https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg	1	miniature_pinscher	0.5603

1. Use the Tweepy library to query additional data via the Twitter API (tweet_json.txt)

In [6]:
```
''' #This code for those who have account in twitter API
import tweepy
from tweepy import OAuthHandler
import json
from timeit import default_timer as timer

# Query Twitter API for each tweet in the Twitter archive and save JSON in a text file
# These are hidden to comply with Twitter's API terms and conditions
consumer_key = 'HIDDEN'
consumer_secret = 'HIDDEN'
access_token = 'HIDDEN'
access_secret = 'HIDDEN'

auth = OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_secret)

api = tweepy.API(auth, wait_on_rate_limit=True)

# NOTE TO STUDENT WITH MOBILE VERIFICATION ISSUES:
# df_1 is a DataFrame with the twitter_archive_enhanced.csv file. You may have to
# change line 17 to match the name of your DataFrame with twitter_archive_enhanced.csv
# NOTE TO REVIEWER: this student had mobile verification issues so the following
# Twitter API code was sent to this student from a Udacity instructor
# Tweet IDs for which to gather additional data via Twitter's API
tweet_ids = df_1.tweet_id.values
len(tweet_ids)

# Query Twitter's API for JSON data for each tweet ID in the Twitter archive
count = 0
fails_dict = {}
start = timer()
# Save each tweet's returned JSON as a new line in a .txt file
with open('tweet_json.txt', 'w') as outfile:
    # This loop will likely take 20-30 minutes to run because of Twitter's rate limit
    for tweet_id in tweet_ids:
        count += 1
        print(str(count) + ": " + str(tweet_id))
        try:
            tweet = api.get_status(tweet_id, tweet_mode='extended')
            print("Success")
            json.dump(tweet._json, outfile)
            outfile.write('\n')
```

```
        except tweepy.TweepError as e:
            print("Fail")
            fails_dict[tweet_id] = e
            pass
end = timer()
print(end - start)
print(fails_dict) '''
```

Out[6]:

```
' #This code for those who have account in twitter API\nimport tweepy\nfrom tweepy impor
t OAuthHandler\nimport json\nfrom timeit import default_timer as timer\n\n# Query Twitte
r API for each tweet in the Twitter archive and save JSON in a text file\n# These are hi
dden to comply with Twitter\'s API terms and conditions\nconsumer_key = \'HIDDEN\'\ncons
umer_secret = \'HIDDEN\'\naccess_token = \'HIDDEN\'\naccess_secret = \'HIDDEN\'\n\nauth
= OAuthHandler(consumer_key, consumer_secret)\nauth.set_access_token(access_token, acces
s_secret)\n\napi = tweepy.API(auth, wait_on_rate_limit=True)\n\n# NOTE TO STUDENT WITH M
OBILE VERIFICATION ISSUES:\n# df_1 is a DataFrame with the twitter_archive_enhanced.csv
file. You may have to\n# change line 17 to match the name of your DataFrame with twitter
_archive_enhanced.csv\n# NOTE TO REVIEWER: this student had mobile verification issues s
o the following\n# Twitter API code was sent to this student from a Udacity instructor\n
# Tweet IDs for which to gather additional data via Twitter\'s API\ntweet_ids = df_1.twe
et_id.values\nlen(tweet_ids)\n\n# Query Twitter\'s API for JSON data for each tweet ID i
n the Twitter archive\ncount = 0\nfails_dict = {}\nstart = timer()\n# Save each tweet\'s
returned JSON as a new line in a .txt file\nwith open(\'tweet_json.txt\', \'w\') as outf
ile:\n    # This loop will likely take 20-30 minutes to run because of Twitter\'s rate l
imit\n    for tweet_id in tweet_ids:\n        count += 1\n        print(str(count) + ":
" + str(tweet_id))\n        try:\n            tweet = api.get_status(tweet_id, tweet_mod
e=\'extended\')\n            print("Success")\n            json.dump(tweet._json, outfil
e)\n            outfile.write(\'\n\')\n        except tweepy.TweepError as e:\n
   print("Fail")\n            fails_dict[tweet_id] = e\n            pass\nend = timer()
\nprint(end - start)\nprint(fails_dict) '
```

In [7]:

```
#I used the tweet json file that provided in Udacity because I don't have account in twi
df_list = []

with open(r"C:\Users\HP\.jupyter\dogs\tweet-json.txt",'r') as file: #open file
    lines = file.readlines()
    for line in lines:
        load_line = json.loads(line)
        tweet_id=load_line['id']
        retweet_count=load_line['retweet_count']
        favorite_count=load_line['favorite_count']
        df_list.append({'tweet_id': tweet_id,
                        'retweet_count': retweet_count,
                        'favorite_count': favorite_count})

t_API = pd.DataFrame(df_list, columns = ['tweet_id', 'retweet_count', 'favorite_count'])
t_API.head()
```

Out[7]:

	tweet_id	retweet_count	favorite_count
0	892420643555336193	8853	39467
1	892177421306343426	6514	33819
2	891815181378084864	4328	25461
3	891689557279858688	8964	42908
4	891327558926688256	9774	41048

Assessing Data

In this section, detect and document at least **eight (8) quality issues and two (2) tidiness issue**. You must use **both** visual assessment programmatic assessement to assess the data.

Note: pay attention to the following key points when you access the data.

- You only want original ratings (no retweets) that have images. Though there are 5000+ tweets in the dataset, not all are dog ratings and some are retweets.
- Assessing and cleaning the entire dataset completely would require a lot of time, and is not necessary to practice and demonstrate your skills in data wrangling. Therefore, the requirements of this project are only to assess and clean at least 8 quality issues and at least 2 tidiness issues in this dataset.
- The fact that the rating numerators are greater than the denominators does not need to be cleaned. This unique rating system is a big part of the popularity of WeRateDogs.
- You do not need to gather the tweets beyond August 1st, 2017. You can, but note that you won't be able to gather the image predictions for these tweets since you don't have access to the algorithm used.

Visual Assessment

In [8]:
```
t_archive
```

Out[8]:

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	s
0	892420643555336193	NaN	NaN	2017-08-01 16:23:56 +0000	href="http://twitter.com/download/ip
1	892177421306343426	NaN	NaN	2017-08-01 00:17:27 +0000	href="http://twitter.com/download/ip
2	891815181378084864	NaN	NaN	2017-07-31 00:18:03 +0000	href="http://twitter.com/download/ip
3	891689557279858688	NaN	NaN	2017-07-30 15:58:51 +0000	href="http://twitter.com/download/ip
4	891327558926688256	NaN	NaN	2017-07-29 16:00:24 +0000	href="http://twitter.com/download/ip
...	...	...	...	...	
2351	666049248165822465	NaN	NaN	2015-11-16 00:24:50 +0000	href="http://twitter.com/download/ip
2352	666044226329800704	NaN	NaN	2015-11-16 00:04:52 +0000	href="http://twitter.com/download/ip

2353	666033412701032449	NaN	NaN	2015-11-15 23:21:54 +0000	href="http://twitter.com/download/ip
2354	666029285002620928	NaN	NaN	2015-11-15 23:05:30 +0000	href="http://twitter.com/download/ip
2355	666020888022790149	NaN	NaN	2015-11-15 22:32:08 +0000	href="http://twitter.com/download/ip

2356 rows × 17 columns

In [9]:
```
image_predictions
```

Out[9]:

	tweet_id	jpg_url	img_num	p1	p1
0	666020888022790149	https://pbs.twimg.com/media/CT4udn0WwAA0aMy.jpg	1	Welsh_springer_spaniel	0.4
1	666029285002620928	https://pbs.twimg.com/media/CT42GRgUYAA5iDo.jpg	1	redbone	0.5
2	666033412701032449	https://pbs.twimg.com/media/CT4521TWwAEvMyu.jpg	1	German_shepherd	0.5
3	666044226329800704	https://pbs.twimg.com/media/CT5Dr8HUEAA-lEu.jpg	1	Rhodesian_ridgeback	0.4
4	666049248165822465	https://pbs.twimg.com/media/CT5IQmsXIAAKY4A.jpg	1	miniature_pinscher	0.5
...	...	...	...	...	
2070	891327558926688256	https://pbs.twimg.com/media/DF6hr6BUMAAzZgT.jpg	2	basset	0.5
2071	891689557279858688	https://pbs.twimg.com/media/DF_q7IAWsAEuuN8.jpg	1	paper_towel	0.1
2072	891815181378084864	https://pbs.twimg.com/media/DGBdLU1WsAANxJ9.jpg	1	Chihuahua	0.7
2073	892177421306343426	https://pbs.twimg.com/media/DGGmoV4XsAAUL6n.jpg	1	Chihuahua	0.3
2074	892420643555336193	https://pbs.twimg.com/media/DGKD1-bXoAAIAUK.jpg	1	orange	0.0

2075 rows × 12 columns

In [10]:
```
t_API
```

Out[10]:

	tweet_id	retweet_count	favorite_count
0	892420643555336193	8853	39467
1	892177421306343426	6514	33819
2	891815181378084864	4328	25461
3	891689557279858688	8964	42908
4	891327558926688256	9774	41048
...	...	...	...

2349	666049248165822465	41	111
2350	666044226329800704	147	311
2351	666033412701032449	47	128
2352	666029285002620928	48	132
2353	666020888022790149	532	2535

2354 rows × 3 columns

Programmatic Assessment

In [11]:
```
t_archive.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2356 entries, 0 to 2355
Data columns (total 17 columns):
 #   Column                      Non-Null Count  Dtype  
---  ------                      --------------  -----  
 0   tweet_id                    2356 non-null   int64  
 1   in_reply_to_status_id       78 non-null     float64
 2   in_reply_to_user_id         78 non-null     float64
 3   timestamp                   2356 non-null   object 
 4   source                      2356 non-null   object 
 5   text                        2356 non-null   object 
 6   retweeted_status_id         181 non-null    float64
 7   retweeted_status_user_id    181 non-null    float64
 8   retweeted_status_timestamp  181 non-null    object 
 9   expanded_urls               2297 non-null   object 
 10  rating_numerator            2356 non-null   int64  
 11  rating_denominator          2356 non-null   int64  
 12  name                        2356 non-null   object 
 13  doggo                       2356 non-null   object 
 14  floofer                     2356 non-null   object 
 15  pupper                      2356 non-null   object 
 16  puppo                       2356 non-null   object 
dtypes: float64(4), int64(3), object(10)
memory usage: 313.0+ KB
```

In [12]:
```
t_archive.describe()
```

Out[12]:

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	retweeted_status_id	retweeted_status_user_id	rating
count	2.356000e+03	7.800000e+01	7.800000e+01	1.810000e+02	1.810000e+02	
mean	7.427716e+17	7.455079e+17	2.014171e+16	7.720400e+17	1.241698e+16	
std	6.856705e+16	7.582492e+16	1.252797e+17	6.236928e+16	9.599254e+16	
min	6.660209e+17	6.658147e+17	1.185634e+07	6.661041e+17	7.832140e+05	
25%	6.783989e+17	6.757419e+17	3.086374e+08	7.186315e+17	4.196984e+09	
50%	7.196279e+17	7.038708e+17	4.196984e+09	7.804657e+17	4.196984e+09	
75%	7.993373e+17	8.257804e+17	4.196984e+09	8.203146e+17	4.196984e+09	
max	8.924206e+17	8.862664e+17	8.405479e+17	8.874740e+17	7.874618e+17	

In [13]:
```
t_archive.isnull().sum()
```

Out[13]:
```
tweet_id                       0
in_reply_to_status_id       2278
```

```
in_reply_to_user_id           2278
timestamp                        0
source                           0
text                             0
retweeted_status_id           2175
retweeted_status_user_id      2175
retweeted_status_timestamp    2175
expanded_urls                   59
rating_numerator                 0
rating_denominator               0
name                             0
doggo                            0
floofer                          0
pupper                           0
puppo                            0
dtype: int64
```

In [14]:
```
t_archive.duplicated().sum()
```

Out[14]:
```
0
```

In [15]:
```
t_archive.sample(10)
```

Out[15]:

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	s
1810	676821958043033607	NaN	NaN	2015-12-15 17:51:44 +0000	href="http://twitter.com/download/ip
1162	723688335806480385	NaN	NaN	2016-04-23 01:41:59 +0000	href="http://twitter.com/download/ip
1107	734776360183431168	NaN	NaN	2016-05-23 16:01:50 +0000	href="http://twitter.com/download/ip
869	761745352076779520	NaN	NaN	2016-08-06 02:06:59 +0000	href="http://twitter.com/download/ip
1558	688789766343622656	NaN	NaN	2016-01-17 18:27:32 +0000	href="http://twitter.com/download/ip
1041	743980027717509120	NaN	NaN	2016-06-18 01:33:55 +0000	href="http://twitter.com/download/ip
937	753298634498793472	NaN	NaN	2016-07-13 18:42:44 +0000	href="http://twitter.com/download/ip
241	846505985330044928	NaN	NaN	2017-03-	

				27 23:35:28 +0000	href="http://twitter.com/download/ip
2292	667160273090932737	NaN	NaN	2015-11-19 01:59:39 +0000	href="http://twitter.com/download/ip
1766	678399652199309312	NaN	NaN	2015-12-20 02:20:55 +0000	href="http://twitter.com/download/ip

In [16]:
```
t_archive.rating_denominator.value_counts()
```

Out[16]:
```
10     2333
11        3
50        3
20        2
80        2
70        1
7         1
15        1
150       1
170       1
0         1
90        1
40        1
130       1
110       1
16        1
120       1
2         1
Name: rating_denominator, dtype: int64
```

In [17]:
```
t_archive.rating_numerator.value_counts()
```

Out[17]:
```
12      558
11      464
10      461
13      351
9       158
8       102
7        55
14       54
5        37
6        32
3        19
4        17
2         9
1         9
75        2
15        2
420       2
0         2
80        1
144       1
17        1
26        1
```

```
20      1
121     1
143     1
44      1
60      1
45      1
50      1
99      1
204     1
1776    1
165     1
666     1
27      1
182     1
24      1
960     1
84      1
88      1
Name: rating_numerator, dtype: int64
```

In [18]:
```
t_archive[t_archive['text'].isnull()]
```

Out[18]:

tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	source	text	retweeted_status_id	retweeted_statu

In [19]:
```
t_archive[t_archive['expanded_urls'].isnull()]
```

Out[19]:

	tweet_id	in_reply_to_status_id	in_reply_to_user_id	timestamp	s
30	886267009285017600	8.862664e+17	2.281182e+09	2017-07-15 16:51:35 +0000	href="http://twitter.com/download/ip
55	881633300179243008	8.816070e+17	4.738443e+07	2017-07-02 21:58:53 +0000	href="http://twitter.com/download/ip
64	879674319642796034	8.795538e+17	3.105441e+09	2017-06-27 12:14:36 +0000	href="http://twitter.com/download/ip
113	870726314365509632	8.707262e+17	1.648776e+07	2017-06-02 19:38:25 +0000	href="http://twitter.com/download/ip
148	863427515083354112	8.634256e+17	7.759620e+07	2017-05-13 16:15:35 +0000	href="http://twitter.com/download/ip
179	857214891891077121	8.571567e+17	1.806710e+08	2017-04-26 12:48:51 +0000	href="http://twitter.com/download/ip
185	856330835276025856	NaN	NaN	2017-04-24 02:15:55 +0000	href="http://twitter.com/download/ip
186	856288084350160898	8.562860e+17	2.792810e+08	2017-04-23 23:26:03 +0000	href="http://twitter.com/download/ip
188	855862651834028034	8.558616e+17	1.943518e+08	2017-04-22 19:15:32 +0000	href="http://twitter.com/download/ip
189	855860136149123072	8.558585e+17	1.361572e+07	2017-04-22 19:05:32	href="http://twitter.com/download/ip

				+0000	
218	850333567704068097	8.503288e+17	2.195506e+07	2017-04-07 13:04:55 +0000	href="http://twitter.com/download/ip
228	848213670039564288	8.482121e+17	4.196984e+09	2017-04-01 16:41:12 +0000	href="http://twitter.com/download/ip
234	847617282490613760	8.476062e+17	4.196984e+09	2017-03-31 01:11:22 +0000	href="http://twitter.com/download/ip
274	840698636975636481	8.406983e+17	8.405479e+17	2017-03-11 22:59:09 +0000	href="http://twitter.com/download/ip
290	838150277551247360	8.381455e+17	2.195506e+07	2017-03-04 22:12:52 +0000	href="http://twitter.com/download/ip
291	838085839343206401	8.380855e+17	2.894131e+09	2017-03-04 17:56:49 +0000	href="http://twitter.com/download/ip
313	835246439529840640	8.352460e+17	2.625958e+07	2017-02-24 21:54:03 +0000	href="http://twitter.com/download/ip
342	832088576586297345	8.320875e+17	3.058208e+07	2017-02-16 04:45:50 +0000	href="http://twitter.com/download/ip
346	831926988323639298	8.319030e+17	2.068372e+07	2017-02-15 18:03:45 +0000	href="http://twitter.com/download/ip
375	828361771580813312	NaN	NaN	2017-02-05 21:56:51 +0000	<a href="http://twitter rel="nofollow"
387	826598799820865537	8.265984e+17	4.196984e+09	2017-02-01 01:11:25 +0000	href="http://twitter.com/download/ip
409	823333489516937216	8.233264e+17	1.582854e+09	2017-01-23 00:56:15 +0000	href="http://twitter.com/download/ip
427	821153421864615936	8.211526e+17	1.132119e+08	2017-01-17 00:33:26 +0000	href="http://twitter.com/download/ip
498	813130366689148928	8.131273e+17	4.196984e+09	2016-12-25 21:12:41 +0000	href="http://twitter.com/download/ip
513	811647686436880384	8.116272e+17	4.196984e+09	2016-12-21 19:01:02 +0000	href="http://twitter.com/download/ip
570	801854953262350336	8.018543e+17	1.185634e+07	2016-11-24 18:28:13 +0000	href="http://twitter.com/download/ip
576	800859414831898624	8.008580e+17	2.918590e+08	2016-11-22 00:32:18 +0000	href="http://twitter.com/download/ip

611	797165961484890113	7.971238e+17	2.916630e+07	2016-11-11 19:55:50 +0000	href="http://twitter.com/download/ip
701	786051337297522688	7.727430e+17	7.305050e+17	2016-10-12 03:50:17 +0000	href="http://twitter.com/download/ip
707	785515384317313025	NaN	NaN	2016-10-10 16:20:36 +0000	href="http://twitter.com/download/ip
843	766714921925144576	7.667118e+17	4.196984e+09	2016-08-19 19:14:16 +0000	href="http://twitter.com/download/ip
857	763956972077010945	7.638652e+17	1.584641e+07	2016-08-12 04:35:10 +0000	href="http://twitter.com/download/ip
967	750381685133418496	7.501805e+17	4.717297e+09	2016-07-05 17:31:49 +0000	href="http://twitter.com/download/ip
1005	747651430853525504	7.476487e+17	4.196984e+09	2016-06-28 04:42:46 +0000	href="http://twitter.com/download/ip
1080	738891149612572673	7.384119e+17	3.589728e+08	2016-06-04 00:32:32 +0000	href="http://twitter.com/download/ip
1295	707983188426153984	7.079801e+17	2.319108e+09	2016-03-10 17:35:20 +0000	href="http://twitter.com/download/ip
1345	704491224099647488	7.044857e+17	2.878549e+07	2016-03-01 02:19:31 +0000	href="http://twitter.com/download/ip
1445	696518437233913856	NaN	NaN	2016-02-08 02:18:30 +0000	href="http://twitter.com/download/ip
1446	696490539101908992	6.964887e+17	4.196984e+09	2016-02-08 00:27:39 +0000	href="http://twitter.com/download/ip
1474	693644216740769793	6.936422e+17	4.196984e+09	2016-01-31 03:57:23 +0000	href="http://twitter.com/download/ip
1479	693582294167244802	6.935722e+17	1.198989e+09	2016-01-30 23:51:19 +0000	href="http://twitter.com/download/ip
1497	692423280028966913	6.924173e+17	4.196984e+09	2016-01-27 19:05:49 +0000	href="http://twitter.com/download/ip
1523	690607260360429569	6.903413e+17	4.670367e+08	2016-01-22 18:49:36 +0000	href="http://twitter.com/download/ip
1598	686035780142297088	6.860340e+17	4.196984e+09	2016-01-10 04:04:10 +0000	href="http://twitter.com/download/ip
1605	685681090388975616	6.855479e+17	4.196984e+09	2016-01-	

				09 04:34:45 +0000	href="http://twitter.com/download/ip
1618	684969860808454144	6.849598e+17	4.196984e+09	2016-01-07 05:28:35 +0000	href="http://twitter.com/download/ip
1663	682808988178739200	6.827884e+17	4.196984e+09	2016-01-01 06:22:03 +0000	href="http://twitter.com/download/ip
1689	681340665377193984	6.813394e+17	4.196984e+09	2015-12-28 05:07:27 +0000	href="http://twitter.com/download/ip
1774	678023323247357953	6.780211e+17	4.196984e+09	2015-12-19 01:25:31 +0000	href="http://twitter.com/download/ip
1819	676590572941893632	6.765883e+17	4.196984e+09	2015-12-15 02:32:17 +0000	href="http://twitter.com/download/ip
1844	675849018447167488	6.758457e+17	4.196984e+09	2015-12-13 01:25:37 +0000	href="http://twitter.com/download/ip
1895	674742531037511680	6.747400e+17	4.196984e+09	2015-12-10 00:08:50 +0000	href="http://twitter.com/download/ip
1905	674606911342424069	6.744689e+17	4.196984e+09	2015-12-09 15:09:55 +0000	href="http://twitter.com/download/ip
1914	674330906434379776	6.658147e+17	1.637468e+07	2015-12-08 20:53:11 +0000	href="http://twitter.com/download/ip
1940	673716320723169284	6.737159e+17	4.196984e+09	2015-12-07 04:11:02 +0000	href="http://twitter.com/download/ip
2038	671550332464455680	6.715449e+17	4.196984e+09	2015-12-01 04:44:10 +0000	href="http://twitter.com/download/ip
2149	669684865554620416	6.693544e+17	4.196984e+09	2015-11-26 01:11:28 +0000	href="http://twitter.com/download/ip
2189	668967877119254528	6.689207e+17	2.143566e+07	2015-11-24 01:42:25 +0000	href="http://twitter.com/download/ip
2298	667070482143944705	6.670655e+17	4.196984e+09	2015-11-18 20:02:51 +0000	href="http://twitter.com/download/ip

In [20]:
```
t_archive.source.value_counts()
```

Out[20]:
```
<a href="http://twitter.com/download/iphone" rel="nofollow">Twitter for iPhone</a>     2
221
<a href="http://vine.co" rel="nofollow">Vine - Make a Scene</a>
91
<a href="http://twitter.com" rel="nofollow">Twitter Web Client</a>
33
<a href="https://about.twitter.com/products/tweetdeck" rel="nofollow">TweetDeck</a>
```

```
11
Name: source, dtype: int64
```

In [21]:
```
t_archive.name.value_counts()
```

Out[21]:
```
None          745
a              55
Charlie        12
Cooper         11
Lucy           11
              ...
Dex             1
Ace             1
Tayzie          1
Grizzie         1
Christoper      1
Name: name, Length: 957, dtype: int64
```

In [22]:
```
image_predictions.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2075 entries, 0 to 2074
Data columns (total 12 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   tweet_id  2075 non-null   int64  
 1   jpg_url   2075 non-null   object 
 2   img_num   2075 non-null   int64  
 3   p1        2075 non-null   object 
 4   p1_conf   2075 non-null   float64
 5   p1_dog    2075 non-null   bool   
 6   p2        2075 non-null   object 
 7   p2_conf   2075 non-null   float64
 8   p2_dog    2075 non-null   bool   
 9   p3        2075 non-null   object 
 10  p3_conf   2075 non-null   float64
 11  p3_dog    2075 non-null   bool   
dtypes: bool(3), float64(3), int64(2), object(4)
memory usage: 152.1+ KB
```

In [23]:
```
image_predictions.describe()
```

Out[23]:

	tweet_id	img_num	p1_conf	p2_conf	p3_conf
count	2.075000e+03	2075.000000	2075.000000	2.075000e+03	2.075000e+03
mean	7.384514e+17	1.203855	0.594548	1.345886e-01	6.032417e-02
std	6.785203e+16	0.561875	0.271174	1.006657e-01	5.090593e-02
min	6.660209e+17	1.000000	0.044333	1.011300e-08	1.740170e-10
25%	6.764835e+17	1.000000	0.364412	5.388625e-02	1.622240e-02
50%	7.119988e+17	1.000000	0.588230	1.181810e-01	4.944380e-02
75%	7.932034e+17	1.000000	0.843855	1.955655e-01	9.180755e-02
max	8.924206e+17	4.000000	1.000000	4.880140e-01	2.734190e-01

In [24]:
```
image_predictions.isnull().sum()
```

Out[24]:
```
tweet_id    0
jpg_url     0
img_num     0
p1          0
p1_conf     0
```

```
p1_dog      0
p2          0
p2_conf     0
p2_dog      0
p3          0
p3_conf     0
p3_dog      0
dtype: int64
```

In [25]:
```
image_predictions.duplicated().sum()
```

Out[25]:
```
0
```

In [26]:
```
image_predictions.sample(10)
```

Out[26]:

	tweet_id	jpg_url	img_num	p1	p1_conf	p
192	669567591774625800	https://pbs.twimg.com/media/CUrlK1DWoAAhECq.jpg	1	Chihuahua	0.980511	
811	692417313023332352	https://pbs.twimg.com/media/CZv13u5WYAA6wQe.jpg	1	bison	0.208922	
1809	832757312314028032	https://pbs.twimg.com/media/C46MWnFVYAUg1RK.jpg	2	Cardigan	0.160888	
1305	753375668877008896	https://pbs.twimg.com/media/CnSHLFeWgAAwV-I.jpg	1	bluetick	0.360071	
795	690938899477221376	https://pbs.twimg.com/media/CZa1QnSWEAAEOVr.jpg	1	geyser	0.370318	
1307	753420520834629632	https://pbs.twimg.com/ext_tw_video_thumb/75342...	1	balloon	0.267961	
122	668221241640230912	https://pbs.twimg.com/media/CUX_rAyWsAYZOQ5.jpg	1	chow	0.395101	
383	673342308415348736	https://pbs.twimg.com/media/CVgxQc5XIAAYL0W.jpg	1	ski_mask	0.981017	
1671	813096984823349248	https://pbs.twimg.com/media/C0izZULWgAAKD-F.jpg	1	Great_Dane	0.128056	
593	679475951516934144	https://pbs.twimg.com/media/CW37xZbUoAAUXe5.jpg	1	Maltese_dog	0.145742	

In [27]:
```
t_API.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2354 entries, 0 to 2353
Data columns (total 3 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   tweet_id        2354 non-null   int64
 1   retweet_count   2354 non-null   int64
 2   favorite_count  2354 non-null   int64
dtypes: int64(3)
memory usage: 55.3 KB
```

In [28]:
```
t_API.describe()
```

Out[28]:

	tweet_id	retweet_count	favorite_count
count	2.354000e+03	2354.000000	2354.000000
mean	7.426978e+17	3164.797366	8080.968564
std	6.852812e+16	5284.770364	11814.771334
min	6.660209e+17	0.000000	0.000000
25%	6.783975e+17	624.500000	1415.000000
50%	7.194596e+17	1473.500000	3603.500000
75%	7.993058e+17	3652.000000	10122.250000

max	8.924206e+17	79515.000000	132810.000000

In [29]:
```
t_API.isnull().sum()
```

Out[29]:
```
tweet_id           0
retweet_count      0
favorite_count     0
dtype: int64
```

In [30]:
```
t_API.duplicated().sum()
```

Out[30]:
```
0
```

In [31]:
```
t_API.sample(10)
```

Out[31]:

	tweet_id	retweet_count	favorite_count
1066	740373189193256964	9220	20648
1081	738537504001953792	1759	5575
1786	677557565589463040	1322	2665
729	781661882474196992	3129	11634
245	845677943972139009	5365	27154
488	813910438903693312	2194	10342
1617	684959798585110529	3487	7004
1998	672488522314567680	482	1187
2287	667174963120574464	88	262
1984	672884426393653248	897	1661

Quality issues

1.These columns (in_reply_to_status_id, in_reply_to_user_id, retweeted_status_id,retweeted_status_user_id, retweeted_status_timestamp) are not needed and contain many null values.

2.timestamp should be converted to data type.

3.The Anchor element in source.

4.We have 55 dogs whose name is a.

5.source should be converted to category data type.

6.We have 59 nulls in expanded_urls.

7.The denominator greater than 10 and less than 10.

8.img_num should be converted to category data type.

9.Underscores in p1 and p2 and p3.

10.Inconsistent capitalization in dog breeds in p1,p2 columns.

11.Some column names are not clear.

12.Drop rating_numerator and rating_denominator.

13.Drop doggo and floofer and pupper,puppo columns.

Tidiness issues

1. In twitter archive dataset we should have one column instead two columns rating_numerator and rating_denominator by divided rating_numerator and rating_denominator
2. In twitter archive dataset we should have one column for dog type instead four columns
3. merge all dataframe in one frame

Cleaning Data

In this section, clean **all** of the issues you documented while assessing.

Note: Make a copy of the original data before cleaning. Cleaning includes merging individual pieces of data according to the rules of tidy data. The result should be a high-quality and tidy master pandas DataFrame (or DataFrames, if appropriate).

In [32]:
```
# Make copies of original pieces of data
t_archive_clean=t_archive.copy()
image_predictions_clean=image_predictions.copy()
t_API_clean=t_API.copy()
```

Quality issues

Issue #1: These columns (in_reply_to_status_id, in_reply_to_user_id, retweeted_status_id retweeted_status_user_id, retweeted_status_timestamp) are not needed and contain many null values

Define

Drop these columns (in_reply_to_status_id, in_reply_to_user_id, retweeted_status_id retweeted_status_user_id, retweeted_status_timestamp)

Code

In [33]:
```
t_archive_clean=t_archive_clean.drop(['in_reply_to_status_id', 'in_reply_to_user_id','re
```

Test

In [34]:
```
t_archive_clean.head()
```

Out[34]:

	tweet_id	timestamp	source	text	
0	892420643555336193	2017-08-01 16:23:56 +0000	<a href="http://twitter.com/download/iphone" r...	This is Phineas. He's a mystical	https://twitter.com/dog_rat

				boy. Only eve...	
1	892177421306343426	2017-08-01 00:17:27 +0000	<a href="http://twitter.com/download/iphone" r...	This is Tilly. She's just checking pup on you....	https://twitter.com/dog_rat
2	891815181378084864	2017-07-31 00:18:03 +0000	<a href="http://twitter.com/download/iphone" r...	This is Archie. He is a rare Norwegian Pouncin...	https://twitter.com/dog_rat
3	891689557279858688	2017-07-30 15:58:51 +0000	<a href="http://twitter.com/download/iphone" r...	This is Darla. She commenced a snooze mid meal...	https://twitter.com/dog_rat
4	891327558926688256	2017-07-29 16:00:24 +0000	<a href="http://twitter.com/download/iphone" r...	This is Franklin. He would like you to stop ca...	https://twitter.com/dog_rat

Issue #2: Timestamp should be converted to data type

Define

Converted timestamp to datatype

Code

In [35]:
```
t_archive_clean['timestamp']=pd.to_datetime(t_archive_clean['timestamp'])
```

Test

In [36]:
```
t_archive_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2356 entries, 0 to 2355
Data columns (total 12 columns):
 #   Column              Non-Null Count  Dtype              
---  ------              --------------  -----              
 0   tweet_id            2356 non-null   int64              
 1   timestamp           2356 non-null   datetime64[ns, UTC]
 2   source              2356 non-null   object             
 3   text                2356 non-null   object             
 4   expanded_urls       2297 non-null   object             
 5   rating_numerator    2356 non-null   int64              
 6   rating_denominator  2356 non-null   int64              
 7   name                2356 non-null   object             
 8   doggo               2356 non-null   object             
 9   floofer             2356 non-null   object             
 10  pupper              2356 non-null   object             
 11  puppo               2356 non-null   object             
dtypes: datetime64[ns, UTC](1), int64(3), object(8)
memory usage: 221.0+ KB
```

Issue #3: The Anchor element in source

Define

Remove a element from source

Code

In [37]:
```
import re
t_archive_clean.source=t_archive_clean.source.apply(lambda x:re.sub('<[^<]+?>','',x))
```

Test

In [38]:
```
t_archive_clean.source.value_counts()
```

Out[38]:
```
Twitter for iPhone     2221
Vine - Make a Scene      91
Twitter Web Client       33
TweetDeck                11
Name: source, dtype: int64
```

Issue #4: We have 55 dogs whose name is a

Define

Replace a in name to None

Code

In [39]:
```
t_archive_clean.name=t_archive_clean.name.str.replace('a','None')
```

Test

In [40]:
```
t_archive_clean.query('name =="a"')
```

Out[40]:

tweet_id	timestamp	source	text	expanded_urls	rating_numerator	rating_denominator	name	doggo	floofe

Issue #5: Source should be converted to category data type

Define

Change source column to category data type

Code

In [41]:
```
t_archive_clean.source=t_archive_clean.source.astype("category")
```

Test

In [42]:
```
t_archive_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2356 entries, 0 to 2355
Data columns (total 12 columns):
```

```
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   tweet_id             2356 non-null   int64
 1   timestamp            2356 non-null   datetime64[ns, UTC]
 2   source               2356 non-null   category
 3   text                 2356 non-null   object
 4   expanded_urls        2297 non-null   object
 5   rating_numerator     2356 non-null   int64
 6   rating_denominator   2356 non-null   int64
 7   name                 2356 non-null   object
 8   doggo                2356 non-null   object
 9   floofer              2356 non-null   object
 10  pupper               2356 non-null   object
 11  puppo                2356 non-null   object
dtypes: category(1), datetime64[ns, UTC](1), int64(3), object(7)
memory usage: 205.1+ KB
```

Issue #6: We have 59 nulls in expanded_urls

Define

Drop 59 nulls values in expanded_urls

Code

In [43]:
```
t_archive_clean.dropna(subset=['expanded_urls'],inplace=True)
```

In [44]:
```
t_archive_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2297 entries, 0 to 2355
Data columns (total 12 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   tweet_id             2297 non-null   int64
 1   timestamp            2297 non-null   datetime64[ns, UTC]
 2   source               2297 non-null   category
 3   text                 2297 non-null   object
 4   expanded_urls        2297 non-null   object
 5   rating_numerator     2297 non-null   int64
 6   rating_denominator   2297 non-null   int64
 7   name                 2297 non-null   object
 8   doggo                2297 non-null   object
 9   floofer              2297 non-null   object
 10  pupper               2297 non-null   object
 11  puppo                2297 non-null   object
dtypes: category(1), datetime64[ns, UTC](1), int64(3), object(7)
memory usage: 217.8+ KB
```

Issue #7: The rating_denominator greater than 10 and less than 10

Define

Drop rating_denominator greater than 10 and less than 10

Code

In [45]:
```
d_10=t_archive_clean.query('rating_denominator >10').index
t_archive_clean.drop(d_10,inplace=True)
```

In [46]:
```
d_10=t_archive_clean.query('rating_denominator <10').index
t_archive_clean.drop(d_10,inplace=True)
```

Test

In [47]:
```
t_archive_clean.rating_denominator.value_counts()
```

Out[47]:
```
10    2278
Name: rating_denominator, dtype: int64
```

Issue #8: img_num should be converted to category data type

Define

Change img_num to category data type

Code

In [48]:
```
image_predictions_clean.img_num=image_predictions_clean.img_num.astype("category")
```

Test

In [49]:
```
image_predictions_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2075 entries, 0 to 2074
Data columns (total 12 columns):
 #   Column    Non-Null Count  Dtype   
---  ------    --------------  -----   
 0   tweet_id  2075 non-null   int64   
 1   jpg_url   2075 non-null   object  
 2   img_num   2075 non-null   category
 3   p1        2075 non-null   object  
 4   p1_conf   2075 non-null   float64 
 5   p1_dog    2075 non-null   bool    
 6   p2        2075 non-null   object  
 7   p2_conf   2075 non-null   float64 
 8   p2_dog    2075 non-null   bool    
 9   p3        2075 non-null   object  
 10  p3_conf   2075 non-null   float64 
 11  p3_dog    2075 non-null   bool    
dtypes: bool(3), category(1), float64(3), int64(1), object(4)
memory usage: 138.1+ KB
```

Issue #9: Underscores in p1 and p2 and p3

Define

Remove underscores in p1 and p2 and p3

Code

In [50]:
```
image_predictions_clean['p1']=image_predictions_clean['p1'].str.replace('_',' ')
image_predictions_clean['p2']=image_predictions_clean['p2'].str.replace('_',' ')
image_predictions_clean['p3']=image_predictions_clean['p3'].str.replace('_',' ')
```

Test

In [51]:
```
image_predictions_clean[['p1','p2','p3']]
```

Out[51]:

	p1	p2	p3
0	Welsh springer spaniel	collie	Shetland sheepdog
1	redbone	miniature pinscher	Rhodesian ridgeback
2	German shepherd	malinois	bloodhound
3	Rhodesian ridgeback	redbone	miniature pinscher
4	miniature pinscher	Rottweiler	Doberman
...	...	...	...
2070	basset	English springer	German short-haired pointer
2071	paper towel	Labrador retriever	spatula
2072	Chihuahua	malamute	kelpie
2073	Chihuahua	Pekinese	papillon
2074	orange	bagel	banana

2075 rows × 3 columns

Issue #10: Inconsistent capitalization in dog breeds in p1 and p2, p3 columns

Define

fixing inconsistent capitalization in dog breeds in p1 and p2, p3 columns

Code

In [52]:
```
image_predictions_clean['p1']=image_predictions_clean['p1'].str.title()
image_predictions_clean['p2']=image_predictions_clean['p2'].str.title()
image_predictions_clean['p3']=image_predictions_clean['p3'].str.title()
```

Test

In [53]:
```
image_predictions_clean[['p1','p2','p3']]
```

Out[53]:

	p1	p2	p3
0	Welsh Springer Spaniel	Collie	Shetland Sheepdog
1	Redbone	Miniature Pinscher	Rhodesian Ridgeback
2	German Shepherd	Malinois	Bloodhound
3	Rhodesian Ridgeback	Redbone	Miniature Pinscher
4	Miniature Pinscher	Rottweiler	Doberman
...	...	...	...
2070	Basset	English Springer	German Short-Haired Pointer
2071	Paper Towel	Labrador Retriever	Spatula
2072	Chihuahua	Malamute	Kelpie

2073	Chihuahua	Pekinese	Papillon
2074	Orange	Bagel	Banana

2075 rows × 3 columns

Issue #11: Some column names are not clear

Define

Code

In [54]:
```
t_archive_clean.rename(columns={'name':'dog_name'},inplace=True)
```

In [55]:
```
image_predictions_clean.rename(columns={'jpg_url':'img_url'},inplace=True)
```

Test

In [56]:
```
t_archive_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2278 entries, 0 to 2355
Data columns (total 12 columns):
 #   Column              Non-Null Count  Dtype              
---  ------              --------------  -----              
 0   tweet_id            2278 non-null   int64              
 1   timestamp           2278 non-null   datetime64[ns, UTC]
 2   source              2278 non-null   category           
 3   text                2278 non-null   object             
 4   expanded_urls       2278 non-null   object             
 5   rating_numerator    2278 non-null   int64              
 6   rating_denominator  2278 non-null   int64              
 7   dog_name            2278 non-null   object             
 8   doggo               2278 non-null   object             
 9   floofer             2278 non-null   object             
 10  pupper              2278 non-null   object             
 11  puppo               2278 non-null   object             
dtypes: category(1), datetime64[ns, UTC](1), int64(3), object(7)
memory usage: 216.0+ KB
```

In [57]:
```
image_predictions.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2075 entries, 0 to 2074
Data columns (total 12 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   tweet_id  2075 non-null   int64  
 1   jpg_url   2075 non-null   object 
 2   img_num   2075 non-null   int64  
 3   p1        2075 non-null   object 
 4   p1_conf   2075 non-null   float64
 5   p1_dog    2075 non-null   bool   
 6   p2        2075 non-null   object 
 7   p2_conf   2075 non-null   float64
 8   p2_dog    2075 non-null   bool   
 9   p3        2075 non-null   object 
 10  p3_conf   2075 non-null   float64
 11  p3_dog    2075 non-null   bool   
dtypes: bool(3), float64(3), int64(2), object(4)
memory usage: 152.1+ KB
```

Tidiness issues

Issue #1: In twitter archive dataset we should have one column instead two columns rating_numerator and rating_denominator by divided rating_numerator and rating_denominator

Define

Create column for rating that contain rating_numerator / rating_denominator

Code

In [58]:
```
t_archive_clean['rating']=t_archive_clean['rating_numerator']/t_archive_clean['rating_de
```

Test

In [59]:
```
t_archive_clean['rating']
```

Out[59]:
```
0       1.3
1       1.3
2       1.2
3       1.3
4       1.2
       ...
2351    0.5
2352    0.6
2353    0.9
2354    0.7
2355    0.8
Name: rating, Length: 2278, dtype: float64
```

Issue #2: In twitter archive dataset we should have one column for dog type instead four columns

Define

Create dog type column insted four column(doggo,floofer,pupper,puppo)

Code

In [60]:
```
t_archive_clean['dog_type']=t_archive_clean.text.str.extract('(doggo|floofer|pupper|pupp
```

Test

In [61]:
```
t_archive_clean.dog_type.value_counts()
```

Out[61]:
```
pupper     257
doggo       89
puppo       34
floofer      4
Name: dog_type, dtype: int64
```

Quality issues

Issue #12: Drop rating_numerator and rating_denominator

Define

Drop rating_numerator and rating_denominator by drop function

Code

In [62]:
```
t_archive_clean=t_archive_clean.drop(['rating_numerator','rating_denominator'],axis=1)
```

Test

In [63]:
```
t_archive_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2278 entries, 0 to 2355
Data columns (total 12 columns):
 #   Column          Non-Null Count  Dtype              
---  ------          --------------  -----              
 0   tweet_id        2278 non-null   int64              
 1   timestamp       2278 non-null   datetime64[ns, UTC]
 2   source          2278 non-null   category           
 3   text            2278 non-null   object             
 4   expanded_urls   2278 non-null   object             
 5   dog_name        2278 non-null   object             
 6   doggo           2278 non-null   object             
 7   floofer         2278 non-null   object             
 8   pupper          2278 non-null   object             
 9   puppo           2278 non-null   object             
 10  rating          2278 non-null   float64            
 11  dog_type        384 non-null    object             
dtypes: category(1), datetime64[ns, UTC](1), float64(1), int64(1), object(8)
memory usage: 216.0+ KB
```

Issue #13: Drop doggo and floofer and pupper,puppo columns.

Define

Drop doggo and floofer and pupper,puppo columns by drop function

Code

In [64]:
```
t_archive_clean.drop(['doggo','floofer','pupper','puppo'],axis=1,inplace=True)
```

Test

In [65]:
```
t_archive_clean
```

Out[65]:

	tweet_id	timestamp	source	text	expanded_urls	d
0	892420643555336193	2017-08-01 16:23:56+00:00	Twitter for iPhone	This is Phineas. He's a mystical boy. Only eve...	https://twitter.com/dog_rates/status/892420643...	Phi
1	892177421306343426	2017-08-01 00:17:27+00:00	Twitter for	This is Tilly. She's just	https://twitter.com/dog_rates/status/892177421...	

			iPhone	checking pup on you....		
2	891815181378084864	2017-07-31 00:18:03+00:00	Twitter for iPhone	This is Archie. He is a rare Norwegian Pouncin...	https://twitter.com/dog_rates/status/891815181...	
3	891689557279858688	2017-07-30 15:58:51+00:00	Twitter for iPhone	This is Darla. She commenced a snooze mid meal...	https://twitter.com/dog_rates/status/891689557...	DNo
4	891327558926688256	2017-07-29 16:00:24+00:00	Twitter for iPhone	This is Franklin. He would like you to stop ca...	https://twitter.com/dog_rates/status/891327558...	FrN
...	...	...	...	...	...	
2351	666049248165822465	2015-11-16 00:24:50+00:00	Twitter for iPhone	Here we have a 1949 1st generation vulpix. Enj...	https://twitter.com/dog_rates/status/666049248...	
2352	666044226329800704	2015-11-16 00:04:52+00:00	Twitter for iPhone	This is a purebred Piers Morgan. Loves to Netf...	https://twitter.com/dog_rates/status/666044226...	
2353	666033412701032449	2015-11-15 23:21:54+00:00	Twitter for iPhone	Here is a very happy pup. Big fan of well-main...	https://twitter.com/dog_rates/status/666033412...	
2354	666029285002620928	2015-11-15 23:05:30+00:00	Twitter for iPhone	This is a western brown Mitsubishi terrier. Up...	https://twitter.com/dog_rates/status/666029285...	
2355	666020888022790149	2015-11-15 22:32:08+00:00	Twitter for iPhone	Here we have a Japanese Irish Setter. Lost eye...	https://twitter.com/dog_rates/status/666020888...	

2278 rows × 8 columns

Tidiness issues

Issue #3: Merge all dataframes in one frame

Define

Merge all dataframes in one frame by merge function

Code

In [66]:
```
df_archive_API=pd.merge(t_archive_clean,t_API_clean,on='tweet_id',how='inner')
```

In [67]:
```
df_master=pd.merge(df_archive_API,image_predictions_clean,on='tweet_id',how='inner')
```

Test

In [68]:
```
df_master.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2055 entries, 0 to 2054
Data columns (total 21 columns):
 #   Column          Non-Null Count  Dtype              
---  ------          --------------  -----              
 0   tweet_id        2055 non-null   int64              
 1   timestamp       2055 non-null   datetime64[ns, UTC]
 2   source          2055 non-null   category           
 3   text            2055 non-null   object             
 4   expanded_urls   2055 non-null   object             
 5   dog_name        2055 non-null   object             
 6   rating          2055 non-null   float64            
 7   dog_type        330 non-null    object             
 8   retweet_count   2055 non-null   int64              
 9   favorite_count  2055 non-null   int64              
 10  img_url         2055 non-null   object             
 11  img_num         2055 non-null   category           
 12  p1              2055 non-null   object             
 13  p1_conf         2055 non-null   float64            
 14  p1_dog          2055 non-null   bool               
 15  p2              2055 non-null   object             
 16  p2_conf         2055 non-null   float64            
 17  p2_dog          2055 non-null   bool               
 18  p3              2055 non-null   object             
 19  p3_conf         2055 non-null   float64            
 20  p3_dog          2055 non-null   bool               
dtypes: bool(3), category(2), datetime64[ns, UTC](1), float64(4), int64(3), object(8)
memory usage: 283.4+ KB
```

Storing Data

Save gathered, assessed, and cleaned master dataset to a CSV file named "twitter_archive_master.csv".

In [69]:
```
df_master.to_csv("twitter_archive_master.csv",index=False)
```

In [70]:
```
df_master
```

Out[70]:

	tweet_id	timestamp	source	text	expanded_urls	d
0	892420643555336193	2017-08-01 16:23:56+00:00	Twitter for iPhone	This is Phineas. He's a mystical boy. Only eve...	https://twitter.com/dog_rates/status/892420643...	Phi
1	892177421306343426	2017-08-01 00:17:27+00:00	Twitter for iPhone	This is Tilly. She's just checking	https://twitter.com/dog_rates/status/892177421...	

				pup on you....		
2	891815181378084864	2017-07-31 00:18:03+00:00	Twitter for iPhone	This is Archie. He is a rare Norwegian Pouncin...	https://twitter.com/dog_rates/status/891815181...	
3	891689557279858688	2017-07-30 15:58:51+00:00	Twitter for iPhone	This is Darla. She commenced a snooze mid meal...	https://twitter.com/dog_rates/status/891689557...	DNo
4	891327558926688256	2017-07-29 16:00:24+00:00	Twitter for iPhone	This is Franklin. He would like you to stop ca...	https://twitter.com/dog_rates/status/891327558...	FrN
...	...	...	...	...	...	
2050	666049248165822465	2015-11-16 00:24:50+00:00	Twitter for iPhone	Here we have a 1949 1st generation vulpix. Enj...	https://twitter.com/dog_rates/status/666049248...	
2051	666044226329800704	2015-11-16 00:04:52+00:00	Twitter for iPhone	This is a purebred Piers Morgan. Loves to Netf...	https://twitter.com/dog_rates/status/666044226...	
2052	666033412701032449	2015-11-15 23:21:54+00:00	Twitter for iPhone	Here is a very happy pup. Big fan of well-main...	https://twitter.com/dog_rates/status/666033412...	
2053	666029285002620928	2015-11-15 23:05:30+00:00	Twitter for iPhone	This is a western brown Mitsubishi terrier. Up...	https://twitter.com/dog_rates/status/666029285...	
2054	666020888022790149	2015-11-15 22:32:08+00:00	Twitter for iPhone	Here we have a Japanese Irish Setter. Lost eye...	https://twitter.com/dog_rates/status/666020888...	

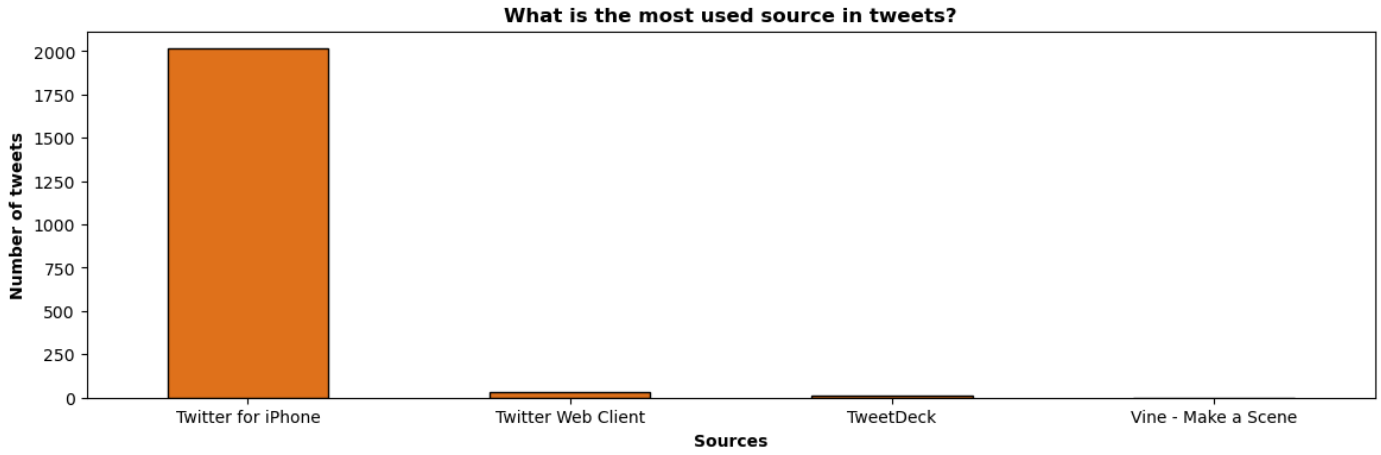
2055 rows × 21 columns

Analyzing and Visualizing Data

In this section, analyze and visualize your wrangled data. You must produce at least **three (3) insights and one (1) visualization.**

Question 1: What is the most used source in tweets?

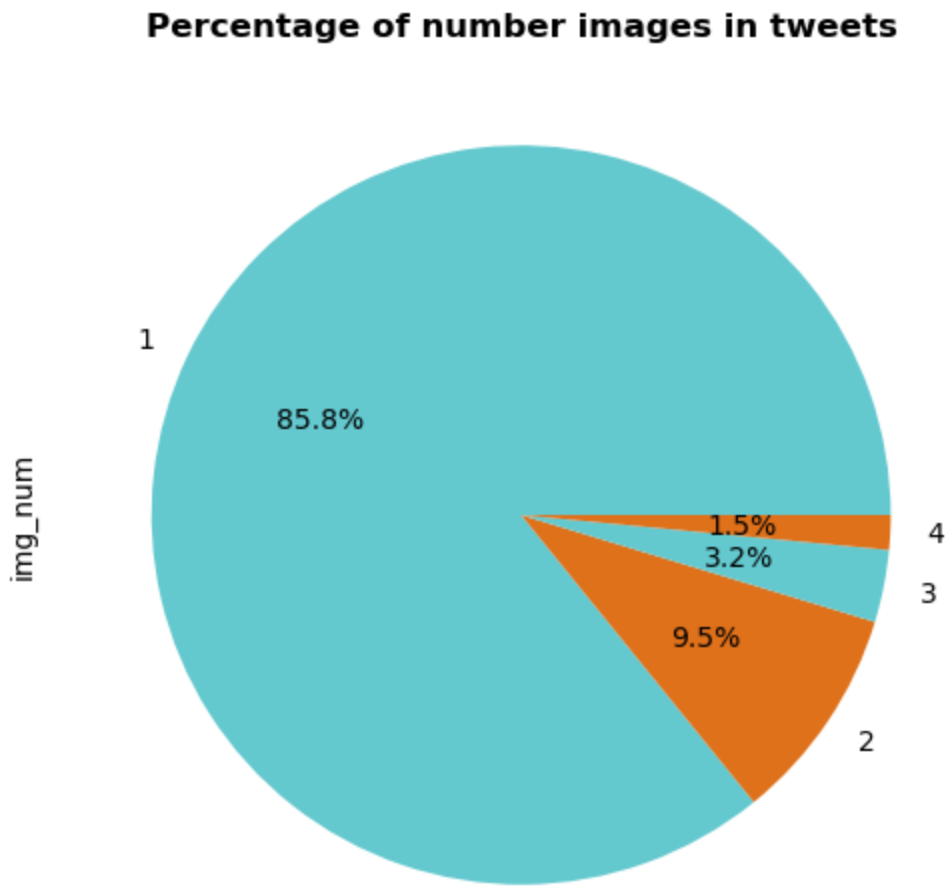
In [78]:
```
v_source=df_master.source.value_counts()
v_source.plot(kind='bar',edgecolor='black',rot=0,figsize=[14,4],color=['#DF711B'])
plt.title("What is the most used source in tweets?",weight='bold')
plt.ylabel("Number of tweets",weight='bold')
plt.xlabel("Sources",weight='bold');
```



Most tweeted from `Twitter for iPhone`

Question 2: What is the percentage of number images in tweets?

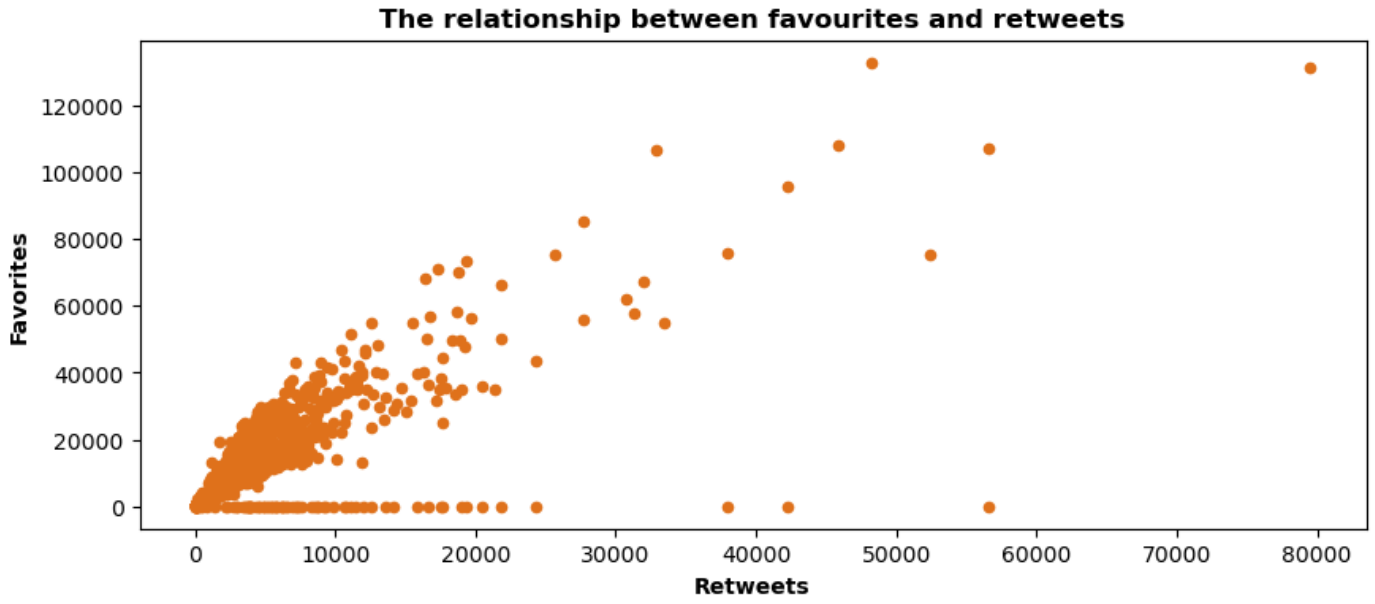
In [97]:
```
df_master.img_num.value_counts().plot.pie(colors=['#64C9CF','#DF711B'],figsize=(8,6),aut
plt.title('Percentage of number images in tweets',weight='bold');
```



85% of tweets have one image in the tweet

Question 3: What is the relationship between favourites and retweets is positive correlation?

In [98]:
```
df_master.plot(kind='scatter',x='retweet_count',y='favorite_count',rot=0,figsize=[10,4],
plt.title("The relationship between favourites and retweets",weight='bold')
plt.ylabel("Favorites",weight='bold')
plt.xlabel("Retweets",weight='bold');
```



The relationship between favourites and retweets is positive correlation

Question 4: What is the most rating and mean of rating?

In [99]:
```
df_master.rating.mean()
```

Out[99]:
```
1.1705596107055949
```

In [100…
```
df_master.rating.value_counts()
```

Out[100]:
```
1.2      473
1.0      429
1.1      413
1.3      283
0.9      150
0.8       95
0.7       51
1.4       40
0.5       34
0.6       32
0.3       19
0.4       15
0.2        9
0.1        4
0.0        2
2.6        1
2.7        1
177.6      1
7.5        1
1.5        1
42.0       1
Name: rating, dtype: int64
```

We have most rating is 1.4 with 473 tweet and mean is 1.1705596107055949.

Question 5: What is the best type of dog?

In [101…
```
df_master.dog_type.value_counts()
```

Out[101]:
```
pupper     224
doggo       74
puppo       29
floofer      3
Name: dog_type, dtype: int64
```

The most dog type is pupper.

Sources

https://www.pythontutorial.net/python-regex/python-regex-sub/

https://stackoverflow.com/questions/45999415/removing-html-tags-in-pandas

https://www.geeksforgeeks.org/drop-rows-from-the-dataframe-based-on-certain-condition-applied-on-a-column/

https://towardsdatascience.com/how-to-merge-pandas-dataframes-35afe8b1497c

https://pandas.pydata.org/docs/reference/api/pandas.Series.str.extract.html